A 3.33Gb/s (1200,720) Low-Density Parity Check Code Decoder

Chien-Ching Lin, Kai-Li Lin, Hsie-Chia Chang and Chen-Yi Lee

Department of Electronics Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C.
cclin@si2lab.org

Abstract:

In this paper, a (1200,720) LDPC decoder based on an irregular parity check matrix is presented. For achieving higher chip density and less critical path delay, the proposed architecture features a new data reordering such that only one specific data bus exists between message memories and computational units. Moreover, the LDPC decoder can also process two different codewords concurrently to increase throughput and datapath efficiency. After chip implementation, a 3.33Gb/s data rate is achieved with 8 decoding iterations in the 21.23mm² 0.18μm silicon area. The other 0.13μm chip with the 10.24mm² core can further reach a 5.92Gb/s data rate under 1.02V supply.

1. Introduction

Lowdensity paritycheck (LDPC) code first introduced by Gallager [1] has engaged much research interest after its rediscovery [2]. The sparse property of parity check matrix H makes the decoding algorithm simple and practical at good communication rates [2]. And its performance can be designed to approach Shannon limits with large block lengths like turbo code[3]. However, LDPC decoders based on sumproduct algorithm (SPA) are capable of parallel implementation that leads to a much higher decoding speed than turbo decoders. Hence the inherent dominance of LDPC code can enhance system performances for high speed wireless communications.

LDPC code is often represented by bipartite graph [4] where N bit nodes and M check nodes are connected by edges according to $M \times N$ matrix H . Fig. 1 shows an example with 10 bit nodes and 6 check nodes. The column weight of H determines the number of edges (or degree) for each bit node to connect to check nodes, while the row weight of H determines the connections for each check node. A LDPC code with equal weights for columns and rows is a regular code and otherwise is termed irregular. It has been shown that irregular codes can outperform those based on regular graphs [5]. However, the irregularity causes signal congestion problems during VLSI implementation and requires more area for routing wires.

The fully parallel implementation in [6] demands large area to accommodate interconnections and has only 50% chip density. The partially parallel architecture also suffers from the same complications, and may be even worse due to multiplexers. Furthermore, the critical path delay

Figure 1: The bipartite graph of (10,4) LDPC code.

induced by global routing damages the maximum achievable throughput. In this paper, a more efficient method that schedules the data flow and eliminates multiplexers for less signal routing is proposed and applied to a (1200,720) LDPC decoder. The implementation results show the decoder can achieve above 70% chip density and 3.33Gb/s throughput with 8 decoding iterations.

This paper is organized as follows. The decoding algorithm is described in section 2.. And section 3. presents the decoder architecture. The chip implementation is shown in section 4.. Finally, the conclusion is also given in section 5..

2. Decoding Algorithm

The decoding of LDPC codes is based on sum-product algorithm (SPA) [2], or belief propagation (BP) algorithm, which iteratively updates the *a posteriori* probabilities of bit nodes. In logarithmic domain, the probability values are often represented by log-likelihood ratio (LLR) $L(x) = \log \frac{P\{x=0\}}{P\{x=1\}}$ for less computational complexity. The decoding procedure consists of two steps, horizontal step (H-step) and vertical step (V-step). In horizontal step, the message sent from check node c_j to bit node b_i can be expressed by

$$L_{c_j \rightarrow b_i} = 2 \tanh^{-1} \left(\prod_{b_{i'} \in \mathcal{B}(j) \setminus b_i} \tanh \left(\frac{L_{b_{i'} \rightarrow c_j}}{2} \right) \right), \quad (1)$$

where $\mathcal{B}(j) \setminus b_i$ is the set of bit nodes connected check node c_j except b_i . On the other hand, the message sent from bit node b_i to check node c_j in vertical step can be

expressed by

$$L_{b_i \rightarrow c_j} = L(b_i) + \sum_{c_j \in \Psi(i) \setminus c_j} L_{c_j \rightarrow b_i}. \quad (2)$$

where $\Psi(i) \setminus c_j$ is the set of check nodes connected to b_i except c_j , and $L(b_i)$ is the channel value. The circuit to implement (1) can be a table look approach [6]. An alternative is the sub-optimal expression of (1) which is

$$L_{c_j \rightarrow b_i} \approx \left(\prod_{b_{j'} \in \mathcal{B}(j) \setminus b_i} \text{sign}(L_{b_{j'} \rightarrow c_j}) \right) \min_{b_{j'} \in \mathcal{B}(j) \setminus b_i} (|L_{b_{j'} \rightarrow c_j}|) \quad (3)$$

This approximation results in a significant reduction of hardware complexity but causes a little performance degradation. The iterative decoding flow chart is illustrated in Fig. 2, where the syndrome check identifies the valid codeword by calculating M parity checks from estimated data.

Figure 2: Iterative decoding flowchart for LDPC codes.

3. Architecture Design

In Fig. 2, Hstep and Vstep are linked in a loop to decode valid codewords iteratively. The computational units for Hstep and Vstep are check node unit (CNU) and bit node unit (BNU). Since the partially parallel architecture is adopted for less computational units, each iteration requires 4 cycles by dividing the 480×1200 parity check matrix H into four 240×600 submatrixes, which is shown in Fig. 3.

Figure 3: The submatrixes of H .

The proposed LDPC decoder architecture illustrated in Fig. 4 contains the input buffer, 240 CNUs, 600 BNUs, and two dedicated message memory units (MMU). The set of data processed by CNUs are $\{h_{00}, h_{01}\}$ and $\{h_{10}, h_{11}\}$, whereas the data fed into BNUs should be $\{h_{00}, h_{10}\}$ and $\{h_{01}, h_{11}\}$. Note that MMUs can reorder the data sequence such that only one specific data bus exists between MMUs and datapaths. Hence many multiplexers can be eliminated in contrast with other approaches. In MMU-1, sub-blocks B , C , D , and E capture the outputs from CNUs while subblocks A and C deliver the message data to BNU

Figure 4: The proposed LDPC decoder architecture.

after reordering. The extra sub-block E is used as temporal memory for reducing the interconnection between other sub-blocks. And the MMU-0 is analogous to MMU-1 but in a reverse direction.

Fig. 5 shows the detailed timing diagram of reordering data sequence in MMU-1. The inputs of BNUs sequentially appear in sub-blocks A and C after reordering the data from CNUs. The solution to switch data sequence also enables the decoder to process two different codewords concurrently without stalls. Therefore, the LDPC decoder is not only area-efficient but the decoding speed is compatible with the fully parallel architecture in [6]. The other components in Fig. 4 will be detailed in the following subsections.

Figure 5: The timing diagram of storage sub-blocks in MMU-1.

3.1 Input buffer

Input buffer is also a storage component that receives and keeps channel values for iterative decoding. Channel values should be fed into CNUs during initialization and BNUs while performing V-step. For this design, the input buffer is connected to BNU and provides channel values

for CNU through BNU and MMU0, leading to less signal connections. In order to eliminate multiplexers in partially parallel architecture, the buffer structure is proposed in Fig. 6. The buffer is divided to four sections where

Figure 6: The architecture of input buffer.

buf0 is a shift register that serially receives the channel values and buf3 transports the associated channel values to BNU.

(a) Channel values

(b) Timing diagram

Figure 7: The timing diagram of input buffer.

The timing diagram of input buffer is presented in Fig. 7. Channel values of two different codewords are shifted within the buffer ring as shown in Fig. 6. Therefore, only buf3 is connected to BNU, leading to a much simple signal transfer.

3.2 Check Node Unit

Based on (3), the CNU can be implemented by a sorter that search the minimum magnitude. The sorter can be further modified to enhance the decoding speed by simultaneously updating all edges connected to the same check node. Fig. 8 illustrates the proposed CNU architecture where all the 6bit inputs are represented as sign-magnitude notation. The CNU can be divided into two parts: one is 1bit sign-multiplication and the other is 5-bit sorter that search the minimum value and the second minimum value from the inputs. The new message for

each bit node is a combination of the sign bit according to (3) and the new magnitude which is either "min" or "2nd min" of the sorter.

Figure 8: The CNU architecture.

3.3 Bit Node Unit

Fig. 9 shows the block diagram of BNU. According to (2), BNU receives the channel value and the message values linked to the same bit node. All inputs with sign-magnitude (SM) notation are firstly converted to be 2's complement (TC) representation, and then summed to perform the updated calculation. The summed values are also clipped to avoid overflow.

Figure 9: The BNU architecture.

Note that if C1, C2, and C3 are set to zero, the channel value input will be directly bypassed to the outputs of BNU. This will produce a path for channel values to reach CNUs to complete initialization that requires four clock cycles.

4. Chip Implementation

A test chip has been fabricated in a 1.8V, 0.18μm 1P6M CMOS technology. The chip size is 25 mm² while the core occupies 21.23mm². The total gate count is 1.15M including two MMUs while the chip core density is about 71.2%. After static timing analysis (STA) and postlayout simulation, the decoder achieves 3.33Gb/s throughput with 8 decoding iterations under 1.62V supply and worst speed corner. The estimation also includes crosstalk analysis for signal wires that cause coupling noise. Furthermore, the (1200,720) LDPC code has been applied to MBOFDM UWB system [7]. The fixedpoint performance in AWGN channel is also shown in Fig. 10. The word lengths of

channel value and message value are 5 bits and 6 bits respectively.

Figure 10: The bit error rate (BER) curves in MBQFDM UWB system.

A second test chip is implemented in a 1.2V, 0.13 μ m 1P8M CMOS technology. The maximum decoding speed has been improved to 5.92Gb/s with 8 decoding iterations. And the chip size becomes 13.5mm² where the core constitutes 10.24mm². Moreover, the chip density grows to about 75.4% because of two extra metal layers. Fig. 11 shows the 0.13 μ m chip layout and the distribution of MMUs. The uniformity of MMU distribution minimizes the routing congestion issues particularly. Table 1 summarizes the LDPC decoder chips in this paper and makes a comparison with the fully parallel design in [6].

Figure 11: The 0.13 μ m chip layout.

5. Conclusion

The high speed and area efficient LDPC decoder is presented. The message memories architecture permits parallel decoding of two codewords and diminishes the routing congestion issues. In addition, the data rescheduling minimizes the signal routing between datapaths and memory units. Consequently, the chip becomes smaller due to the

Table 1: Summary of the LDPC decoder chips.

	Chip 1	Chip 2	[6]
Block length	1200	1200	1024
Technology	0.13 μ m	0.18 μ m	0.16 μ m
Data rate	8 iters. 5.92Gb/s ¹ 64 iters. 820Mb/s ¹	3.33Gb/s ² 461Mb/s ²	N.A. 512Mb/s
Power consumption	268mW @1.2V	644mW @1.8V	690mW @1.5V
Gate count	1.15M	1.15M	1.75M
Chip size	13.5mm ²	25mm ²	52.5mm ²
Chip density	75.4%	71.2%	50%

¹ 1.02V supply

² 1.62V supply

the increased chip density. After implementation in 0.18 μ m technology, the chip can achieve the 3.33Gb/s data rate with 8 decoding iterations. Furthermore, the 0.13 μ m chip reaches the maximum 5.92Gb/s data rate with 13.5mm² area and 268mW power consumption.

6. Acknowledgments

The authors are grateful to the support from the National Science Council, R.O.C. under grant NSC 93-2220-E-009-033 and from the Ministry of Economic Affairs, R.O.C. under grant 93-EC-17-A-03-S1-005. This work is in part supported from NCTUMediaTek project.

References:

- [1] R. G. Gallager, *Low-Density Parity-Check Codes*. MA: MIT Press, 1963.
- [2] D. J. C. MacKay, "Good errorcorrecting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.
- [3] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanik, "Design of capacityapproaching irregular low-density paritycheck codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.
- [4] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. IT27, no. 5, pp. 399–431, Sept. 1981.
- [5] M. G. Luby, M. Mitzenmacher, M. A. Shokollahi, and D. A. Spielman, "Improved lowdensity paritycheck codes using irregular graphs," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 585–598, Feb. 2001.
- [6] A. Blanksby and C. Howland, "A 690mw 1Gb/s 1024b, rate1/2 lowdensity paritycheck code decoder," *IEEE J. Solid-State Circuits*, vol. 37, no. 3, pp. 404–412, Mar. 2002.
- [7] A. Batra *et al.*, "Multiband OFDM physical layer proposal for IEEE 802.15 task group 3a," submitted to IEEE P802.15 working group for WPANs, Sept. 2004.